H 5

```
FFFFFFFFFFFFFFF         111              111        XXX         XXX
FFFFFFFFFFFFFFF         111              111        XXX         XXX
FFFFFFFFFFFFFFF         111              111        XXX         XXX
FFF                  111111           111111        XXX         XXX
FFF                  111111           111111        XXX         XXX
FFF                  111111           111111        XXX         XXX
FFF                     111              111           XXX   XXX
FFF                     111              111           XXX   XXX
FFF                     111              111           XXX   XXX
FFFFFFFF.FFF            111              111              XXX
FFFFFFFFFFFF            111              111              XXX
FFFFFFFFFFFF            111              111              XXX
FFF                     111              111           XXX   XXX
FFF                     111              111           XXX   XXX
FFF                     111              111           XXX   XXX
FFF                     111              111        XXX         XXX
FFF                     111              111        XXX         XXX
FFF                     111              111        XXX         XXX
FFF                  111111111        111111111     XXX         XXX
FFF                  111111111        111111111     XXX         XXX
FFF                  111111111        111111111     XXX         XXX
```

```
_$25
Symb
----
IOC$
IO_C
IO_C
IO_D
IO_F
IO_S
KILL

KILL
KILL
LB_E
LB_D
LB_F
LB_H
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CO
L_CO
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP
MARK
MARK
MARK
MARK
MARK
```

FILEID**DIRSCN

```
DDDDDDDD      IIIIII    RRRRRRRR      SSSSSSSS    CCCCCCCC  NN      NN
DDDDDDDD      IIIIII    RRRRRRRR      SSSSSSSS    CCCCCCCC  NN      NN
DD      DD      II      RR      RR  SS          CC          NN      NN
DD      DD      II      RR      RR  SS          CC          NN      NN
DD      DD      II      RR      RR  SS          CC          NNNN    NN
DD      DD      II      RR      RR  SS          CC          NNNN    NN
DD      DD      II      RRRRRRRR      SSSSSS    CC          NN  NN  NN
DD      DD      II      RRRRRRRR        SSSSSS  CC          NN  NN  NN
DD      DD      II      RR  RR                SS  CC        NN    NNNN
DD      DD      II      RR  RR                SS  CC        NN    NNNN
DD      DD      II      RR    RR              SS  CC        NN      NN      ....
DD      DD      II      RR    RR              SS  CC        NN      NN      ....
DDDDDDDD      IIIIII    RR      RR  SSSSSSSS      CCCCCCCC  NN      NN      ....
DDDDDDDD      IIIIII    RR      RR  SSSSSSSS      CCCCCCCC  NN      NN      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE DIRSCN (
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-000'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1
;     7          0007  1 !
;     8          0008  1 !**********************************************************************
;     9          0009  1 !*                                                                    *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                             *
;    13          0013  1 !*                                                                    *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    19          0019  1 !*  TRANSFERRED.                                                       *
;    20          0020  1 !*                                                                    *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    23          0023  1 !*  CORPORATION.                                                       *
;    24          0024  1 !*                                                                    *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    27          0027  1 !*                                                                    *
;    28          0028  1 !*                                                                    *
;    29          0029  1 !**********************************************************************
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 !
;    33          0033  1 ! FACILITY:  F11ACP Structure Level 2
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       This routine performs the basic directory scan, searching for the
;    38          0038  1 !       given entry.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:
;    41          0041  1 !
;    42          0042  1 !       STARLET operating system, including privileged system services
;    43          0043  1 !       and internal exec routines.
;    44          0044  1 !
;    45          0045  1 !--
;    46          0046  1 !
;    47          0047  1 !
;    48          0048  1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  30-Dec-1977  11:14
;    49          0049  1 !
;    50          0050  1 ! MODIFIED BY:
;    51          0051  1 !
;    52          0052  1 !       V03-009 CDS0006         Christian D. Saether    29-Aug-1984
;    53          0053  1 !               Establish last block for search in directory scan also.
;    54          0054  1 !
;    55          0055  1 !       V03-008 CDS0005         Christian D. Saether     5-Aug-1984
;    56          0056  1 !               Fix directory index search still.
;    57          0057  1 !
```

```
;    58          0058  1 ;       V03-007 CDS0004         Christian D. Saether     5-Aug-1984
;    59          0059  1 ;               Fix sense of length extended compare in directory
;    60          0060  1 ;               index search.  Make dirindx cell size 15 bytes.
;    61          0061  1 ;
;    62          0062  1 ;       V03-006 CDS0003         Christian D. Saether     2-Aug-1984
;    63          0063  1 ;               Add support for revamped directory index cache.
;    64          0064  1 ;
;    65          0065  1 ;       V03-005 CDS0002         Christian D. Saether    25-Apr-1984
;    66          0066  1 ;               Remove references to DIRIDX in the FCB.
;    67          0067  1 ;
;    68          0068  1 ;       V03-004 ACG0408         Andrew C. Goldstein,    23-Mar-1984  11:17
;    69          0069  1 ;               Make rest of global storage based
;    70          0070  1 ;
;    71          0071  1 ;       V03-003 CDS0002         Christian D. Saether    29-Dec-1983
;    72          0072  1 ;               Use L_NORM linkage and BIND_COMMON macro.
;    73          0073  1 ;
;    74          0074  1 ;       V03-002 CDS0001         Christian D. Saether    12-Dec-1983
;    75          0075  1 ;               Move GLOBAL data declaration to COMMON module.
;    76          0076  1 ;
;    77          0077  1 ;       V03-001 LMP0080         L. Mark Pilant,         15-Feb-1983  12:26
;    78          0078  1 ;               Add support for propagation of file attributes.  This
;    79          0079  1 ;               consists of remembering the FID of the highest version
;    80          0080  1 ;               of the file of the same name and type.
;    81          0081  1 ;
;    82          0082  1 ;       V02-008 ACG0259         Andrew C. Goldstein,    27-Jan-1982  20:16
;    83          0083  1 ;               Fix counting of entries when skipping records
;    84          0084  1 ;
;    85          0085  1 ;       V02-006 ACG0208         Andrew C. Goldstein,    26-Oct-1981  16:27
;    86          0086  1 ;               Add support for segmented directory records.
;    87          0087  1 ;
;    88          0088  1 ;       V02-005 ACG0167         Andrew C. Goldstein,    16-Apr-1980  19:25
;    89          0089  1 ;               Previous revision history moved to F11B.REV
;    90          0090  1 ;**
;    91          0091  1
;    92          0092  1
;    93          0093  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    94          0094  1 REQUIRE 'SRC$:FCPDEF.B32';
;    95          1085  1
;    96          1086  1
;    97          1087  1 FORWARD ROUTINE
;    98          1088  1         DIR_SCAN        : L_NORM,       ! directory scanner
;    99          1089  1         NEXT_REC        : L_NORM,       ! get next directory record
;   100          1090  1         UPDATE_INDX     : NOVALUE,      ! update directory index entry
;   101          1091  1         NEXT_DIR_REC    : L_NORM;       ! get next matching directory record
```

```
;  103          1092  1 GLOBAL ROUTINE DIR_SCAN (NAME_DESC, FILE_ID, START_BLOCK, START_REC, START_VER, START_PRED, REC_COUNT)
;  104          1093  1         : L_NORM =
;  105          1094  1
;  106          1095  1 !++
;  107          1096  1 !
;  108          1097  1 ! FUNCTIONAL DESCRIPTION:
;  109          1098  1 !
;  110          1099  1 !       This routine scans a directory, searching for the given entry.
;  111          1100  1 !
;  112          1101  1 !
;  113          1102  1 ! CALLING SEQUENCE:
;  114          1103  1 !       DIR_SCAN (ARG1, ARG2, ARG3, ARG4, ARG5, ARG6, ARG7)
;  115          1104  1 !
;  116          1105  1 ! INPUT PARAMETERS:
;  117          1106  1 !       ARG1: address of file name descriptor block
;  118          1107  1 !       ARG2: address of file ID block
;  119          1108  1 !       ARG3: relative block number to start search
;  120          1109  1 !       ARG4: address of record at which to start
;  121          1110  1 !       ARG5: address of version entry at which to start
;  122          1111  1 !       ARG6: address of predecessor record
;  123          1112  1 !       ARG7: maximum number of records to scan
;  124          1113  1 !             (functions only with FIND_FID and a non-matching FID)
;  125          1114  1 !
;  126          1115  1 ! IMPLICIT INPUTS:
;  127          1116  1 !       LAST_ENTRY: name string of last record in previous block
;  128          1117  1 !
;  129          1118  1 ! OUTPUT PARAMETERS:
;  130          1119  1 !       NONE
;  131          1120  1 !
;  132          1121  1 ! IMPLICIT OUTPUTS:
;  133          1122  1 !       DIR_VBN: relative block + 1 of current directory buffer
;  134          1123  1 !       DIR_BUFFER: address of current directory block buffer
;  135          1124  1 !       DIR_RECORD: record number within block of found entry
;  136          1125  1 !       DIR_ENTRY: address in buffer of found record
;  137          1126  1 !       DIR_VERSION: address if buffer of found version entry
;  138          1127  1 !       DIR_PRED: predecessor record to record found
;  139          1128  1 !       LAST_ENTRY: name string of last record in previous block
;  140          1129  1 !       VERSION_LIMIT: version limit of last file name processed
;  141          1130  1 !       VERSION_COUNT: number of versions of current file name passed
;  142          1131  1 !
;  143          1132  1 ! ROUTINE VALUE:
;  144          1133  1 !       1 if entry found
;  145          1134  1 !       0 if no match, in which case:
;  146          1135  1 !               DIR_ENTRY = next record in collating sequence
;  147          1136  1 !                         = 0 if whole directory scanned (name belongs off the end)
;  148          1137  1 !               DIR_VERSION = next version in collating sequence if name & type matched
;  149          1138  1 !                           = 0 if name or type did not match
;  150          1139  1 !
;  151          1140  1 ! SIDE EFFECTS:
;  152          1141  1 !       directory blocks read
;  153          1142  1 !       directory index in FCB updated
;  154          1143  1 !
;  155          1144  1 !--
;  156          1145  1
;  157          1146  2 BEGIN
;  158          1147  2
;  159          1148  2 MAP
```

```
;  160     1149  2          NAME_DESC       : REF BBLOCK;   ! name descriptor block arg
;  161     1150  2
;  162     1151  2      LABEL
;  163     1152  2          DIRINDX_SCAN,                   ! block to search directory index
;  164     1153  2          SEARCH_LOOP;                    ! body of search code
;  165     1154  2
;  166     1155  2      LINKAGE
;  167     1156  2          L_MATCH_NAME    = JSB (REGISTER = 2, REGISTER = 3, REGISTER = 4, REGISTER = 5)
;  168     1157  2                          : NOTUSED (10, 11);
;  169     1158  2
;  170     1159  2      LOCAL
;  171     1160  2          DIRINDX         : REF BBLOCK FIELD (DIRC), ! directory index cache
;  172     1161  2          STATUS,                         ! routine return status
;  173     1162  2          COUNT,                          ! entry count within current block
;  174     1163  2          BLOCK,                          ! relative block number
;  175     1164  2          LAST_BLOCK,                     ! last block of directory to read
;  176     1165  2          MATCHED,                        ! flag indicating name match encountered
;  177     1166  2          DN              : REF BBLOCK,   ! address of name descriptor block
;  178     1167  2          ENTRY           : REF BBLOCK,   ! pointer to current directory record
;  179     1168  2          P               : REF BBLOCK,   ! pointer to current directory version
;  180     1169  2          PREV_ENTRY      : REF BBLOCK;   ! pointer to previous record
;  181     1170  2
;  182     1171  2      BIND_COMMON;
;  183     1172  2
;  184     1173  2      DIR_CONTEXT_DEF;                    ! define directory context fields
;  185     1174  2
;  186     1175  2      EXTERNAL ROUTINE
;  187     1176  2          READ_BLOCK      : L_NORM,       ! read a disk block
;  188     1177  2          MARK_DIRTY      : L_NORM,       ! mark buffer for write back
;  189     1178  2          FMG$MATCH_NAME  : L_MATCH_NAME; ! match general wild card string
;  190     1179  2
;  191     1180  2
;  192     1181  2      ! Initialize basic pointers. Compute the cluster factor of the directory
;  193     1182  2      ! index from the directory size.
;  194     1183  2      !
;  195     1184  2
;  196     1185  2      DN = .NAME_DESC;
;  197     1186  2      STATUS = 0;
;  198     1187  2      BLOCK = .START_BLOCK;
;  199     1188  2      ENTRY = .START_REC;
;  200     1189  2      P = .START_VER;
;  201     1190  2      PREV_ENTRY = .START_PRED;
;  202     1191  2      COUNT = .DIR_RECORD;
;  203     1192  2      MATCHED = 0;
;  204     1193  2
;  205     1194  2      IF .BLOCK GTRU .DIR_FCB[FCB$L_EFBLK]
;  206     1195  2      THEN BLOCK = .DIR_FCB[FCB$L_EFBLK];
;  207     1196  2
;  208     1197  3      SEARCH_LOOP: BEGIN                  ! outer directory search loop
;  209     1198  3
;  210     1199  3      ! Initialize the last block for the search to be the end of file.
;  211     1200  3      !
;  212     1201  3
;  213     1202  3      LAST_BLOCK = .DIR_FCB[FCB$L_EFBLK] - 1;
;  214     1203  3
;  215     1204  4      IF NOT (   CH$RCHAR (.DN[FND_STRING]) EQL '*'
;  216     1205  4              OR CH$RCHAR (.DN[FND_STRING]) EQL '%'
```

```
;  217          1206  4          OR .DN[FND_FIND_FID]
;  218          1207  4          )
;  219          1208  3 AND (DIRINDX = .DIR_FCB [FCB$L_DIRINDX]) NEQ 0
;  220          1209  3 THEN
;  221          1210  3
;  222          1211  3 ! There is a directory index.  This cache contains the last entry
;  223          1212  3 ! of blocks that have already been read.  It allows us to move the
;  224          1213  3 ! starting block for the search into the directory file instead
;  225          1214  3 ! of always starting at the beginning of the file, and limit the
;  226          1215  3 ! end of the search also.
;  227          1216  3 !
;  228          1217  3
;  229          1218  3 DIRINDX_SCAN:
;  230          1219  4     BEGIN
;  231          1220  4     LOCAL
;  232          1221  4          CELL_ADDR,
;  233          1222  4          CELLSIZE,
;  234          1223  4          CMPSIZE,
;  235          1224  4          FNDSTRNG,
;  236          1225  4          PTR,
;  237          1226  4          SEARCH_CELL     : WORD;
;  238          1227  4
;  239          1228  4     IF .DIRINDX [DIRC$W_INUSE] EQL 0
;  240          1229  4     THEN
;  241          1230  4          LEAVE DIRINDX_SCAN;
;  242          1231  4
;  243          1232  4     SEARCH_CELL = .BLOCK/.DIRINDX [DIRC$W_BLKSPERCELL];
;  244          1233  4
;  245          1234  4     CELLSIZE = CMPSIZE = .DIRINDX [DIRC$W_CELLSIZE];
;  246          1235  4
;  247          1236  4     FNDSTRNG = .DN [FND_STRING];
;  248          1237  4
;  249          1238  4     IF .DN [FND_COUNT] LSSU .CMPSIZE
;  250          1239  4     THEN
;  251          1240  4          CMPSIZE = .DN [FND_COUNT];
;  252          1241  4
;  253          1242  4     IF (PTR = CH$FIND_CH (.CMPSIZE, .FNDSTRNG, '*')) NEQ 0
;  254          1243  4     THEN
;  255          1244  4          CMPSIZE = .PTR - .FNDSTRNG;
;  256          1245  4
;  257          1246  4     IF (PTR = CH$FIND_CH (.CMPSIZE, .FNDSTRNG, '%')) NEQ 0
;  258          1247  4     THEN
;  259          1248  4          CMPSIZE = .PTR - .FNDSTRNG;
;  260          1249  4
;  261          1250  4     CELL_ADDR = DIRINDX [DIRC$T_FIRSTCELL]
;  262          1251  4                 + (.SEARCH_CELL)*(.CELLSIZE);
;  263          1252  4
;  264          1253  4 !
;  265          1254  4
;  266          1255  4     UNTIL .SEARCH_CELL GEQU .DIRINDX [DIRC$W_INUSE]
;  267          1256  4     DO
;  268          1257  4          CASE CH$COMPARE (.CMPSIZE, .FNDSTRNG,
;  269          1258  4                           .CELLSIZE, .CELL_ADDR,
;  270          1259  4                           0)
;  271          1260  4               FROM -1 TO 1 OF
;  272          1261  4          SET
;  273          1262  4
```

```
;  274          1263  4 ! Directory index cell is greater than the string we are searching with.
;  275          1264  4 ! Drop out of the loop.
;  276          1265  4 !
;  277          1266  4
;  278          1267  4         [-1]:   EXITLOOP;
;  279          1268  4
;  280          1269  4 ! Exact match.  Drop out of the loop.  We must start the search
;  281          1270  4 ! at the start of this group.
;  282          1271  4 !
;  283          1272  4
;  284          1273  4         [0]:    EXITLOOP;
;  285          1274  4
;  286          1275  4 ! The search string is greater than this directory index entry.  It
;  287          1276  4 ! therefore cannot be in this group or any that we have passed.
;  288          1277  4 ! Update the starting block to reflect that fact, bump cell pointers
;  289          1278  4 ! and keep searching.
;  290          1279  4 !
;  291          1280  4
;  292          1281  5         [1]:    BEGIN
;  293          1282  5                 BLOCK = (.SEARCH_CELL)*(.DIRINDX [DIRC$W_BLKSPERCELL]);
;  294          1283  5                 SEARCH_CELL = .SEARCH_CELL + 1;
;  295          1284  5                 CELL_ADDR = .CELL_ADDR + .CELLSIZE;
;  296          1285  4                 END;
;  297          1286  4         TES;
;  298          1287  4
;  299          1288  4 ! The second part of the search is to establish the end block for the
;  300          1289  4 ! scan.  This time we want the fill character for the compare to
;  301          1290  4 ! extend the potentially shorter search string with hex FF instead
;  302          1291  4 ! of zero so that possible wildcards do not make us stop short.
;  303          1292  4 ! If we exceed the number of cells in use without finding a directory
;  304          1293  4 ! index entry greater than our search string, LAST_BLOCK will remain
;  305          1294  4 ! set to end of file.
;  306          1295  4 !
;  307          1296  4
;  308          1297  4     UNTIL .SEARCH_CELL GEQU .DIRINDX [DIRC$W_INUSE]
;  309          1298  4     DO
;  310          1299  4         CASE CH$COMPARE (.CMPSIZE, .FNDSTRNG,
;  311          1300  4                          .CELLSIZE, .CELL_ADDR,
;  312          1301  4                          -1)
;  313          1302  4             FROM -1 TO 1 OF
;  314          1303  4         SET
;  315          1304  4
;  316          1305  4 ! Directory index cell is greater than the string we are searching with.
;  317          1306  4 ! The entry cannot be beyond here.  Reset the LAST_BLOCK variable to
;  318          1307  4 ! limit the search.
;  319          1308  4 !
;  320          1309  4
;  321          1310  5         [-1]:   BEGIN
;  322          1311  5                 LOCAL
;  323          1312  5                     TEMP;
;  324          1313  5
;  325          1314  5                 TEMP = (.SEARCH_CELL + 1)*(.DIRINDX [DIRC$W_BLKSPERCELL]) - 1;
;  326          1315  5
;  327          1316  5                 IF .TEMP LSSU .LAST_BLOCK
;  328          1317  5                 THEN
;  329          1318  5                     LAST_BLOCK = .TEMP;
;  330          1319  5
```

```
;   331          1320  5                  EXITLOOP;
;   332          1321  4                  END;
;   333          1322  4
;   334          1323  4 ! The search string is equal to or greater than the directory entry.
;   335          1324  4 ! Continue scanning the directory index.
;   336          1325  4 !
;   337          1326  4
;   338          1327  5          [0,1]:  BEGIN
;   339          1328  5                  SEARCH_CELL = .SEARCH_CELL + 1;
;   340          1329  5                  CELL_ADDR = .CELL_ADDR + .CELLSIZE;
;   341          1330  4                  END;
;   342          1331  4
;   343          1332  4          TES;
;   344          1333  4
;   345          1334  3      END;
;   346          1335  3
;   347          1336  3 ! If checking against EOF and the directory index has changed the starting
;   348          1337  3 ! block number, discard the starting record pointers, which are now irrelevant.
;   349          1338  3 !
;   350          1339  3
;   351          1340  3 IF .BLOCK NEQ .START_BLOCK
;   352          1341  3 THEN
;   353          1342  4      BEGIN
;   354          1343  4      ENTRY = 0;
;   355          1344  4      P = 0;
;   356          1345  4      COUNT = 0;
;   357          1346  4      PREV_ENTRY = 0;
;   358          1347  4      LAST_ENTRY[0] = 0;
;   359          1348  3      END;
;   360          1349  3
;   361          1350  3 ! Loop, scanning blocks of the directory until we hit EOF.
;   362          1351  3 !
;   363          1352  3
;   364          1353  3 WHILE 1 DO
;   365          1354  4      BEGIN
;   366          1355  4
;   367          1356  4      IF .BLOCK GTRU .LAST_BLOCK
;   368          1357  4      THEN LEAVE SEARCH_LOOP;
;   369          1358  4      IF .ENTRY EQL 0
;   370          1359  4      THEN
;   371          1360  5          BEGIN
;   372          1361  5          ENTRY = READ_BLOCK (.BLOCK+.DIR_FCB[FCB$L_STLBN],
;   373          1362  5                              .LAST_BLOCK - .BLOCK + 1,
;   374          1363  5                              DIRECTORY_TYPE);
;   375          1364  5          DIR_BUFFER = .ENTRY;
;   376          1365  4          END;
;   377          1366  4
;   378          1367  4 ! Loop, scanning the records of the directory. A record size of -1 indicates
;   379          1368  4 ! the end of the block. We attempt to match name and type against the entry,
;   380          1369  4 ! under control of the various name control flags.
;   381          1370  4 !
;   382          1371  4
;   383          1372  4      WHILE 1
;   384          1373  4      DO
;   385          1374  5          BEGIN
;   386          1375  5          IF .ENTRY[DIR$W_SIZE] EQL 65535
;   387          1376  5          THEN
```

```
;  388     1377  6              BEGIN
;  389     1378  6              IF .PREV_ENTRY NEQ 0
;  390     1379  6              THEN CH$MOVE (.PREV_ENTRY[DIR$B_NAMECOUNT]+1,
;  391     1380  6                            PREV_ENTRY[DIR$B_NAMECOUNT], LAST_ENTRY);
;  392     1381  6              PREV_ENTRY = 0;
;  393     1382  6              EXITLOOP;
;  394     1383  5              END;
;  395     1384  5
;  396     1385  5      ! Do setup and validation for the record.
;  397     1386  5      !
;  398     1387  5
;  399     1388  5          IF .ENTRY[DIR$W_SIZE] + .ENTRY + 2 GEQA .DIR_BUFFER + 512
;  400     1389  5          OR .ENTRY[DIR$V_TYPE] NEQ DIR$C_FID
;  401     1390  5          OR .ENTRY[DIR$B_NAMECOUNT] GTRU FILENAME_LENGTH
;  402     1391  5          OR .ENTRY[DIR$B_NAMECOUNT] GTRU .ENTRY[DIR$W_SIZE] + 2 - DIR$C_LENGTH - DIR$C_VERSION
;  403     1392  5          THEN ERR_EXIT (SS$_BADIRECTORY);
;  404     1393  5
;  405     1394  5      ! If this is a lookup for lowest version and a name has previously matched,
;  406     1395  5      ! see if the name in the record has changed from the previous record. If
;  407     1396  5      ! so, the previous record has the lowest version. This test is made in
;  408     1397  5      ! a seemingly redundant manner with the name change test below to minimize
;  409     1398  5      ! its actual execution.
;  410     1399  5      !
;  411     1400  5
;  412     1401  5          IF .MATCHED
;  413     1402  5          AND .DN[FND_VERSION] EQL -32768
;  414     1403  6          AND (IF .PREV_ENTRY EQL 0
;  415     1404  6              THEN CH$NEQ (.ENTRY[DIR$B_NAMECOUNT]+1,
;  416     1405  6                           ENTRY[DIR$B_NAMECOUNT],
;  417     1406  6                           .ENTRY[DIR$B_NAMECOUNT]+1,
;  418     1407  6                           LAST_ENTRY)
;  419     1408  6              ELSE CH$NEQ (.ENTRY[DIR$B_NAMECOUNT]+1,
;  420     1409  6                           ENTRY[DIR$B_NAMECOUNT],
;  421     1410  6                           .ENTRY[DIR$B_NAMECOUNT]+1,
;  422     1411  6                           PREV_ENTRY[DIR$B_NAMECOUNT])
;  423     1412  6              )
;  424     1413  5          THEN LEAVE SEARCH_LOOP;
;  425     1414  5
;  426     1415  5      ! Attempt to match the name, using a simple string compare if there are
;  427     1416  5      ! no wild cards, otherwise the general wild card match routine.
;  428     1417  5      !
;  429     1418  5
;  430     1419  5          IF
;  431     1420  6              BEGIN
;  432     1421  6
;  433     1422  6              IF .DN[FND_FIND_FID]
;  434     1423  6              THEN 1
;  435     1424  6
;  436     1425  6              ELSE
;  437     1426  7                  BEGIN
;  438     1427  7                  IF NOT .DN[FND_WILD]
;  439     1428  7                  THEN
;  440     1429  7                      CASE CH$COMPARE (.ENTRY[DIR$B_NAMECOUNT],
;  441     1430  7                                       ENTRY[DIR$T_NAME],
;  442     1431  7                                       .DN[FND_COUNT],
;  443     1432  7                                       .DN[FND_STRING]
;  444     1433  7                                       )
```

```
;  445     1434  7                    FROM -1 TO 1 OF
;  446     1435  7                        SET
;  447     1436  7
;  448     1437  7                        [-1]:   0;      ! no match - dir entry precedes name
;  449     1438  7
;  450     1439  7                        [0]:    1;      ! match
;  451     1440  7
;  452     1441  8                        [1]:    BEGIN   ! no match - dir entry is past name
;  453     1442  8                                P = 0;
;  454     1443  8                                LEAVE SEARCH_LOOP;
;  455     1444  7                                END;
;  456     1445  7                        TES
;  457     1446  7
;  458     1447  7                    ELSE
;  459     1448  7                        FMG$MATCH_NAME (.ENTRY[DIR$B_NAMECOUNT],
;  460     1449  7                                        ENTRY[DIR$T_NAME],
;  461     1450  7                                        .DN[FND_COUNT],
;  462     1451  7                                        .DN[FND_STRING]
;  463     1452  7                                        )
;  464     1453  7                    END
;  465     1454  6                END
;  466     1455  6
;  467     1456  6 ! If the name and type match on a record, loop to process the versions of
;  468     1457  6 ! the record.
;  469     1458  6 !
;  470     1459  6
;  471     1460  5            THEN
;  472     1461  6                BEGIN
;  473     1462  6                IF .P EQL 0
;  474     1463  6                THEN
;  475     1464  7                    BEGIN
;  476     1465  7                    P = .ENTRY + DIR$C_LENGTH + .ENTRY[DIR$B_NAMECOUNT] + 1 AND NOT 1;
;  477     1466  7
;  478     1467  8                    IF (IF .PREV_ENTRY EQL 0
;  479     1468  8                        THEN CH$NEQ (.ENTRY[DIR$B_NAMECOUNT]+1,
;  480     1469  8                                     ENTRY[DIR$B_NAMECOUNT],
;  481     1470  8                                    .ENTRY[DIR$B_NAMECOUNT]+1,
;  482     1471  8                                     LAST_ENTRY)
;  483     1472  8                        ELSE CH$NEQ (.ENTRY[DIR$B_NAMECOUNT]+1,
;  484     1473  8                                     ENTRY[DIR$B_NAMECOUNT],
;  485     1474  8                                    .ENTRY[DIR$B_NAMECOUNT]+1,
;  486     1475  8                                     PREV_ENTRY[DIR$B_NAMECOUNT])
;  487     1476  8                        )
;  488     1477  7                    THEN
;  489     1478  8                        BEGIN
;  490     1479  8                        VERSION_COUNT = 0;
;  491     1480  8                        VERSION_LIMIT = .ENTRY[DIR$W_VERLIMIT];
;  492     1481  7                        END;
;  493     1482  6                    END;
;  494     1483  6
;  495     1484  6                UNTIL .P GEQA .ENTRY + .ENTRY[DIR$W_SIZE] + 2
;  496     1485  6                DO
;  497     1486  7                    BEGIN
;  498     1487  7                    IF NOT .MATCHED
;  499     1488  7                    AND NOT .DN[FND_WILD]
;  500     1489  7                    AND NOT .DN[FND_FIND_FID]
;  501     1490  7                    THEN CH$MOVE (FID$C_LENGTH, P[DIR$W_FID], OLD_VERSION_FID);
```

```
;  502          1491  7                MATCHED = 1;
;  503          1492  7                IF .COUNT GEQU .REC_COUNT THEN LEAVE SEARCH_LOOP;
;  504          1493  7
;  505          1494  7                IF
;  506          1495  8                    BEGIN
;  507          1496  8
;  508          1497  8                    IF .DN[FND_FIND_FID]
;  509          1498  8                    THEN CH$EQL (FID$C_LENGTH, .FILE_ID, FID$C_LENGTH, P[DIR$W_FID])
;  510          1499  8
;  511          1500  8                    ELSE IF .DN[FND_WILD_VER]
;  512          1501  8                    OR .DN[FND_MAX_VER]
;  513          1502  8                    OR .DN[FND_VERSION] EQL -.VERSION_COUNT
;  514          1503  8                    THEN 1
;  515          1504  8
;  516          1505  8                    ELSE IF .DN[FND_VERSION] GTR .P[DIR$W_VERSION]
;  517          1506  8                    THEN
;  518          1507  9                        BEGIN
;  519          1508  9                        IF .DN[FND_FLAGS] EQL 0
;  520          1509  9                        THEN LEAVE SEARCH_LOOP
;  521          1510  9                        ELSE EXITLOOP
;  522          1511  9                        END
;  523          1512  9
;  524          1513  8                    ELSE .DN[FND_VERSION] EQL .P[DIR$W_VERSION]
;  525          1514  8
;  526          1515  8                    END
;  527          1516  8
;  528          1517  7                THEN
;  529          1518  8                    BEGIN
;  530          1519  8                    STATUS = 1;
;  531          1520  8                    LEAVE SEARCH_LOOP;
;  532          1521  7                    END;
;  533          1522  7
;  534          1523  7                P = .P + DIR$C_VERSION;
;  535          1524  7                COUNT = .COUNT + 1;
;  536          1525  7                VERSION_COUNT = .VERSION_COUNT + 1;
;  537          1526  6                END;                            ! end of record scanning loop
;  538          1527  5            END;                                ! end of record processing conditional
;  539          1528  5
;  540          1529  5        ! Set up the next record to process.
;  541          1530  5        !
;  542          1531  5
;  543          1532  5            IF .P EQL 0
;  544          1533  5            THEN P = .ENTRY + DIR$C_LENGTH + .ENTRY[DIR$B_NAMECOUNT] + 1 AND NOT 1;
;  545          1534  5            PREV_ENTRY = .ENTRY;
;  546          1535  5            ENTRY = NEXT_REC (.ENTRY);
;  547          1536  5            COUNT = .COUNT + (.ENTRY-.P) / DIR$C_VERSION;
;  548          1537  5            P = 0;
;  549          1538  4            END;                                ! end of block scanning loop
;  550          1539  4
;  551          1540  4        ! We have tripped out of the record scan loop, either because we reached
;  552          1541  4        ! the end of the block or we ran out the record count. In the latter case
;  553          1542  4        ! (i.e., if this was a position to record number call), we are done.
;  554          1543  4        ! Otherwise update the directory index (causing it to be built on the fly)
;  555          1544  4        ! and read the next block.
;  556          1545  4        !
;  557          1546  4
;  558          1547  4            IF .LAST_ENTRY[0] NEQ 0
```

```
;  559          1548  4      THEN
;  560          1549  4          UPDATE_INDX (.BLOCK, .LAST_ENTRY [0], LAST_ENTRY[1], .DIR_FCB);
;  561          1550  4
;  562          1551  4      BLOCK = .BLOCK + 1;
;  563          1552  4      ENTRY = 0;
;  564          1553  4      P = 0;
;  565          1554  4      COUNT = 0;
;  566          1555  4      IF .REC_COUNT LSSU 63 THEN LEAVE SEARCH_LOOP;
;  567          1556  3      END;                                    ! end of block loop
;  568          1557  3
;  569          1558  2  END;                                        ! end of block SEARCH_LOOP
;  570          1559  2
;  571          1560  2  ! We are done searching the directory, and have either found the
;  572          1561  2  ! desired entry or have passed the point where it should be. If we
;  573          1562  2  ! matched on the name of the previous record, back up to it. Point to
;  574          1563  2  ! the last version in the record if we were searching for lowest
;  575          1564  2  ! version; else point off the end of the record.
;  576          1565  2  !
;  577          1566  2
;  578          1567  2  IF NOT .STATUS
;  579          1568  2  AND .MATCHED
;  580          1569  2  AND .P EQL 0
;  581          1570  2  THEN
;  582          1571  3      BEGIN
;  583          1572  3      IF .PREV_ENTRY NEQ 0
;  584          1573  3      THEN
;  585          1574  4          BEGIN
;  586          1575  4          ENTRY = .PREV_ENTRY;
;  587          1576  4          END
;  588          1577  3      ELSE
;  589          1578  4          BEGIN
;  590          1579  4          COUNT = 0;
;  591          1580  4          BLOCK = .BLOCK - 1;
;  592          1581  4          P = READ_BLOCK (.BLOCK+.DIR_FCB[FCB$L_STLBN], 1, DIRECTORY_TYPE);
;  593          1582  4          DIR_BUFFER = .P;
;  594          1583  4          DO
;  595          1584  5              BEGIN
;  596          1585  5              ENTRY = .P;
;  597          1586  5              P = NEXT_REC (.P);
;  598          1587  5              COUNT = .COUNT + (.P - .ENTRY - (.ENTRY[DIR$B_NAMECOUNT]+1 AND NOT 1)) / DIR$C_VERSION;
;  599          1588  5              END
;  600          1589  4          UNTIL .P[DIR$W_SIZE] EQL 65535;
;  601          1590  3          END;
;  602          1591  3      P = .ENTRY + .ENTRY[DIR$W_SIZE] + 2;
;  603          1592  3      IF .DN[FND_VERSION] EQL -32768
;  604          1593  3      THEN
;  605          1594  4          BEGIN
;  606          1595  4          P = .P - DIR$C_VERSION;
;  607          1596  4          VERSION_COUNT = .VERSION_COUNT - 1;
;  608          1597  4          COUNT = .COUNT - 1;
;  609          1598  4          STATUS = 1;
;  610          1599  3          END;
;  611          1600  2      END;
;  612          1601  2
;  613          1602  2  ! Return the record count and pointer in global storage and return status.
;  614          1603  2  !
;  615          1604  2
```

```
;  616          1605  2 DIR_VBN = .BLOCK + 1;
;  617          1606  2 DIR_RECORD = .COUNT;
;  618          1607  2 DIR_ENTRY = .ENTRY;
;  619          1608  2 DIR_VERSION = .P;
;  620          1609  2 DIR_PRED = .PREV_ENTRY;
;  621          1610  2
;  622          1611  2 RETURN .STATUS;
;  623          1612  2
;  624          1613  1 END;                                        ! end of routine DIR_SCAN

                                                            .TITLE  DIRSCN
                                                            .IDENT  \V04-000\

                                                            .EXTRN  READ_BLOCK, MARK_DIRTY
                                                            .EXTRN  FMG$MATCH_NAME

                                                            .PSECT  $CODE$,NOWRT,2

                                  0BFC 00000                .ENTRY  DIR_SCAN, Save R2,R3,R4,R5,R6,R7,R8,R9,R11   ; 1092
              5E               10   C2 00002                SUBL2   #16, SP
                     00D0      CA   9F 00005                PUSHAB  208(BASE)                                    ; 1169
              59     00DC      CA   9E 00009                MOVAB   220(BASE), R9
                       1A      A9   9F 0000E                PUSHAB  26(R9)                                       ; 1171
                       1C      A9   9F 00011                PUSHAB  28(R9)
              58       04      AC   D0 00014                MOVL    NAME_DESC, DN                                ; 1185
                               7E   D4 00018                CLRL    STATUS                                       ; 1186
                       0C      AC   DD 0001A                PUSHL   START_BLOCK                                  ; 1187
              57       10      AC   D0 0001D                MOVL    START_REC, ENTRY                             ; 1188
                       14      AC   DD 00021                PUSHL   START_VER                                    ; 1189
                       18      AC   DD 00024                PUSHL   START_PRED                                   ; 1190
                     00D8      CA   DD 00027                PUSHL   216(BASE)                                    ; 1191
                               7E   D4 0002B                CLRL    MATCHED                                      ; 1192
              50       20      BE   D0 0002D                MOVL    @32(SP), R0                                  ; 1194
         3C   A0       10      AE   D1 00031                CMPL    BLOCK, 60(R0)
                               05   1B 00036                BLEQU   1$
         10   AE       3C      A0   D0 00038                MOVL    60(R0), BLOCK                                ; 1195
              50       20      BE   D0 0003D  1$:           MOVL    @32(SP), R0                                  ; 1202
  56     3C   A0               01   C3 00041                SUBL3   #1, 60(R0), LAST_BLOCK
              2A       08      B8   91 00046                CMPB    @8(DN), #42                                  ; 1204
                               0A   13 0004A                BEQL    2$
              25       08      B8   91 0004C                CMPB    @8(DN), #37                                  ; 1205
                               04   13 00050                BEQL    2$
  03          68               0B   E1 00052                BBC     #11, (DN), 3$                                ; 1206
                             00BA   31 00056  2$:           BRW     12$
              50       20      BE   D0 00059  3$:           MOVL    @32(SP), R0                                  ; 1208
              54     00B0      C0   D0 0005D                MOVL    176(R0), DIRINDX
                               F2   13 00062                BEQL    2$
                               64   B5 00064                TSTW    (DIRINDX)                                    ; 1228
                               EE   13 00066                BEQL    2$
              50       06      A4   3C 00068                MOVZWL  6(DIRINDX), R0                               ; 1232
  50     10   AE               50   C7 0006C                DIVL3   R0, BLOCK, R0
              5B               50   B0 00071                MOVW    R0, SEARCH_CELL
              55       04      A4   3C 00074                MOVZWL  4(DIRINDX), CMPSIZE                          ; 1234
         30   AE               55   D0 00078                MOVL    CMPSIZE, CELLSIZE
         28   AE       08      A8   D0 0007C                MOVL    8(DN), FNDSTRNG                              ; 1236
              55       04      A8   D1 00081                CMPL    4(DN), CMPSIZE                               ; 1238
```

```
                                        04     1E 00085          BGEQU   4$                                          ;
                         55       04    A8     D0 00087          MOVL    4(DN), CMPSIZE                              ; 1240
          28   BE        55             2A     3A 0008B 4$:      LOCC    #42, CMPSIZE, @FNDSTRNG                     ; 1242
                                        02     12 00090          BNEQ    5$                                          ;
                                        51     D4 00092          CLRL    R1                                          ;
                                        51     D5 00094 5$:      TSTL    PTR                                         ;
                                        05     13 00096          BEQL    6$                                          ;
               55        51       28    AE     C3 00098          SUBL3   FNDSTRNG, PTR, CMPSIZE                      ; 1244
          28   BE        55             25     3A 0009D 6$:      LOCC    #37, CMPSIZE, @FNDSTRNG                     ; 1246
                                        02     12 000A2          BNEQ    7$                                          ;
                                        51     D4 000A4          CLRL    R1                                          ;
                                        51     D5 000A6 7$:      TSTL    PTR                                         ;
                                        05     13 000A8          BEQL    8$                                          ;
               55        51       28    AE     C3 000AA          SUBL3   FNDSTRNG, PTR, CMPSIZE                      ; 1248
                         50             5B     3C 000AF 8$:      MOVZWL  SEARCH_CELL, R0                             ; 1251
                         50       30    AE     C4 000B2          MULL2   CELLSIZE, R0                                ;
                    2C   AE       0C A044      9E 000B6          MOVAB   12(R0)[DIRINDX], CELL_ADDR                  ;
                         64             5B     B1 000BC 9$:      CMPW    SEARCH_CELL, (DIRINDX)                      ; 1255
                                        20     1E 000BF          BGEQU   10$                                         ;
 30   AE        00       28   BE        55     2D 000C1          CMPC5   CMPSIZE, @FNDSTRNG, #0, CELLSIZE, -         ; 1257
                                  2C    BE        000C8                  @CELL_ADDR                                  ;
                                        15     1B 000CA          BLEQU   10$                                         ;
                         50             5B     3C 000CC          MOVZWL  SEARCH_CELL, R0                             ; 1282
                         51       06    A4     3C 000CF          MOVZWL  6(DIRINDX), R1                              ;
          10   AE        50             51     C5 000D3          MULL3   R1, R0, BLOCK                               ;
                                        5B     B6 000D8          INCW    SEARCH_CELL                                 ; 1283
                    2C   AE       30    AE     C0 000DA          ADDL2   CELLSIZE, CELL_ADDR                         ; 1284
                                        DB     11 000DF          BRB     9$                                          ; 1257
                         64             5B     B1 000E1 10$:     CMPW    SEARCH_CELL, (DIRINDX)                      ; 1297
                                        2D     1E 000E4          BGEQU   12$                                         ;
 30   AE   FF   8F       28   BE        55     2D 000E6          CMPC5   CMPSIZE, @FNDSTRNG, #-1, CELLSIZE, -        ; 1301
                                  2C    BE        000EE                  @CELL_ADDR                                  ;
                                        18     1E 000F0          BGEQU   11$                                         ; 1299
                         50             5B     3C 000F2          MOVZWL  SEARCH_CELL, R0                             ; 1314
                                        50     D6 000F5          INCL    R0                                          ;
                         51       06    A4     3C 000F7          MOVZWL  6(DIRINDX), R1                              ;
                         50             51     C4 000FB          MULL2   R1, R0                                      ;
                                        50     D7 000FE          DECL    TEMP                                        ;
                         56             50     D1 00100          CMPL    TEMP, LAST_BLOCK                            ; 1316
                                        0E     1E 00103          BGEQU   12$                                         ;
                         56             50     D0 00105          MOVL    TEMP, LAST_BLOCK                            ; 1318
                                        09     11 00108          BRB     12$                                         ; 1310
                                        5B     B6 0010A 11$:     INCW    SEARCH_CELL                                 ; 1328
                    2C   AE       30    AE     C0 0010C          ADDL2   CELLSIZE, CELL_ADDR                         ; 1329
                                        CE     11 00111          BRB     10$                                         ; 1299
                    0C   AC       10    AE     D1 00113 12$:     CMPL    BLOCK, START_BLOCK                          ; 1340
                                        0B     13 00118          BEQL    13$                                         ;
                                        57     D4 0011A          CLRL    ENTRY                                       ; 1343
                                  04    AE     D4 0011C          CLRL    COUNT                                       ; 1345
                                  08    AE     7C 0011F          CLRQ    PREV_ENTRY                                  ; 1346
                                  18    BE     94 00122          CLRB    @24(SP)                                     ; 1347
                         56       10    AE     D1 00125 13$:     CMPL    BLOCK, LAST_BLOCK                           ; 1356
                                        03     1B 00129          BLEQU   14$                                         ;
                                      01FA     31 0012B          BRW     41$                                         ;
                                        57     D5 0012E 14$:     TSTL    ENTRY                                       ; 1358
                                        22     12 00130          BNEQ    15$                                         ;
                                        02     DD 00132          PUSHL   #2                                          ; 1361
```

```
              50            56      14    AE  C3 00134           SUBL3   BLOCK, LAST_BLOCK, R0                     ; 1362
                                    01    A0  9F 00139           PUSHAB  1(R0)                                     ;
                            50      28    BE  D0 0013C           MOVL    @40(SP), R0                               ; 1361
                            51      18    AE  D0 00140           MOVL    BLOCK, R1                                 ;
                                    30  B041  9F 00144           PUSHAB  @48(R0)[R1]                               ;
                     00000G CF            03  FB 00148           CALLS   #3, READ_BLOCK                            ;
                            57            50  D0 0014D           MOVL    R0, ENTRY                                 ;
                      04    A9            57  D0 00150           MOVL    ENTRY, 4(R9)                              ; 1364
                    FFFF    8F            67  B1 00154  15$:     CMPW    (ENTRY), #65535                           ; 1375
                                          1F  12 00159           BNEQ    17$                                       ;
                                    08    AE  D5 0015B           TSTL    PREV_ENTRY                                ; 1378
                                          14  13 0015E           BEQL    16$                                       ;
              51      08    AE            05  C1 00160           ADDL3   #5, PREV_ENTRY, R1                        ; 1379
                            50            61  9A 00165           MOVZBL  (R1), R0                                  ;
                                          50  D6 00168           INCL    R0                                        ;
              5B      08    AE            05  C1 0016A           ADDL3   #5, PREV_ENTRY, R11                       ; 1380
        18    BE            6B            50  28 0016F           MOVC3   R0, (R11), @24(SP)                        ;
                                    08    AE  D4 00174  16$:     CLRL    PREV_ENTRY                                ; 1381
                                  017F    31 00177               BRW     39$                                       ; 1377
                            50            67  3C 0017A  17$:     MOVZWL  (ENTRY), R0                               ; 1388
                            52      02  A740  9E 0017D           MOVAB   2(ENTRY)[R0], R2                          ;
              51      04    A9 00000200   8F  C1 00182           ADDL3   #512, 4(R9), R1                           ;
                            51            52  D1 0018B           CMPL    R2, R1                                    ;
                                          18  1E 0018E           BGEQU   18$                                       ;
                      07            04    A7  93 00190           BITB    4(ENTRY), #7                              ; 1389
                                          12  12 00194           BNEQ    18$                                       ;
                      50    8F      05    A7  91 00196           CMPB    5(ENTRY), #80                             ; 1390
                                          0B  1A 0019B           BGTRU   18$                                       ;
                            50            0C  C2 0019D           SUBL2   #12, R0                                   ; 1391
  50          05    A7      08            00  ED 001A0           CMPZV   #0, #8, 5(ENTRY), R0                      ;
                                          05  1B 001A6           BLEQU   19$                                       ;
                                  0828    8F  BF 001A8  18$:     CHMU    #2088                                     ; 1392
                                              04 001AC           RET                                               ;
                            2D            6E  E9 001AD  19$:     BLBC    MATCHED, 22$                              ; 1401
                    8000    8F      0C    A8  B1 001B0           CMPW    12(DN), #-32768                           ; 1402
                                          25  12 001B6           BNEQ    22$                                       ;
                                    08    AE  D5 001B8           TSTL    PREV_ENTRY                                ; 1403
                                          0E  12 001BB           BNEQ    20$                                       ;
                      50            05    A7  9A 001BD           MOVZBL  5(ENTRY), R0                              ; 1404
                                          50  D6 001C1           INCL    R0                                        ;
        18    BE      05    A7            50  29 001C3           CMPC3   R0, 5(ENTRY), @24(SP)                     ; 1405
                                          10  11 001C9           BRB     21$                                       ;
                      50            05    A7  9A 001CB  20$:     MOVZBL  5(ENTRY), R0                              ; 1408
                                          50  D6 001CF           INCL    R0                                        ;
              54      08    AE            05  C1 001D1           ADDL3   #5, PREV_ENTRY, R4                        ; 1411
              64      05    A7            50  29 001D6           CMPC3   R0, 5(ENTRY), (R4)                        ;
                                          1F  12 001DB  21$:     BNEQ    25$                                       ;
              30            6B            0B  E0 001DD  22$:     BBS     #11, (DN), 27$                            ; 1422
                            1A      01    A8  E8 001E1           BLBS    1(DN), 26$                                ; 1427
                            50      05    A7  9A 001E5           MOVZBL  5(ENTRY), R0                              ; 1429
 04   A8      00      06    A7            50  2D 001E9           CMPC5   R0, 6(ENTRY), #0, 4(DN), @8(DN)           ; 1430
                                    08    B8     001F0                                                             ;
                                          05  1A 001F2           BGTRU   24$                                       ;
                                          1B  1E 001F4           BGEQU   27$                                       ;
                                  00CD    31 001F6  23$:         BRW     37$                                       ;
                                    0C    AE  D4 001F9  24$:     CLRL    P                                         ; 1442
                                  0129    31 001FC  25$:         BRW     41$                                       ; 1443
```

```
                         53      06    A7  9E 001FF 26$:   MOVAB   6(ENTRY), R3                     ; 1449
                         54      04    A8  7D 00203        MOVQ    4(DN), R4                        ;
                         52      05    A7  9A 00207        MOVZBL  5(ENTRY), R2                     ;
                              0000G    30  0020B           BSBW    FMG$MATCH_NAME                   ;
                         E5            50  E9 0020E        BLBC    R0, 23$                          ;
                                 0C    AE  D5 00211 27$:   TSTL    P                                ; 1462
                                       3A  12 00214        BNEQ    30$                              ;
                         54      05    A7  9E 00216        MOVAB   5(ENTRY), R4                     ; 1465
                         50            64  9A 0021A        MOVZBL  (R4), R0                         ;
                         50      07  A047  9E 0021D        MOVAB   7(R0)[ENTRY], R0                 ;
      0C       AE        50            01  CB 00222        BICL3   #1, R0, P                        ;
                                 08    AE  D5 00227        TSTL    PREV_ENTRY                       ; 1467
                                       0C  12 0022A        BNEQ    28$                              ;
                         50            64  9A 0022C        MOVZBL  (R4), R0                         ; 1468
                                       50  D6 0022F        INCL    R0                               ;
      18       BE        64            50  29 00231        CMPC3   R0, (R4), @24(SP)                ; 1469
                                       0E  11 00236        BRB     29$                              ;
                         50            64  9A 00238 28$:   MOVZBL  (R4), R0                         ; 1472
                                       50  D6 0023B        INCL    R0                               ;
               55    08  AE            05  C1 0023D        ADDL3   #5, PREV_ENTRY, R5               ; 1475
               65        64            50  29 00242        CMPC3   R0, (R4), (R5)                   ;
                                       08  13 00246 29$:   BEQL    30$                              ;
                                 1C    BE  B4 00248        CLRW    @28(SP)                          ; 1479
                    18   A9      02    A7  B0 0024B        MOVW    2(ENTRY), 24(R9)                 ; 1480
                         50            67  3C 00250 30$:   MOVZWL  (ENTRY), R0                      ; 1484
                         50      02  A047  9E 00253        MOVAB   2(R0)[ENTRY], R0                 ;
                         50      0C    AE  D1 00258        CMPL    P, R0                            ;
                                       68  1E 0025C        BGEQU   37$                              ;
                         13            6E  E8 0025E        BLBS    MATCHED, 31$                     ; 1487
                         0F      01    A8  E8 00261        BLBS    1(DN), 31$                       ; 1488
               0B        68            0B  E0 00265        BBS     #11, (DN), 31$                   ; 1489
               5B    0C  AE            02  C1 00269        ADDL3   #2, P, R11                       ; 1490
      014C     CA        6B            06  28 0026E        MOVC3   #6, (R11), 332(BASE)             ;
                         6E            01  D0 00274 31$:   MOVL    #1, MATCHED                      ; 1491
                    1C   AC      04    AE  D1 00277        CMPL    COUNT, REC_COUNT                 ; 1492
                                       3A  1E 0027C        BGEQU   35$                              ;
               0C        68            0B  E1 0027E        BBC     #11, (DN), 32$                   ; 1497
               54    0C  AE            02  C1 00282        ADDL3   #2, P, R4                        ; 1498
               64    08  BC            06  29 00287        CMPC3   #6, @FILE_ID, (R4)               ;
                                       24  11 0028C        BRB     33$                              ;
               22        68            03  E0 0028E 32$:   BBS     #3, (DN), 34$                    ; 1500
               1E        68            09  E0 00292        BBS     #9, (DN), 34$                    ; 1501
                         50      1C    BE  3C 00296        MOVZWL  @28(SP), R0                      ; 1502
                         50            50  CE 0029A        MNEGL   R0, R0                           ;
  50  0C       A8        10            00  EC 0029D        CMPV    #0, #16, 12(DN), R0              ;
                                       0F  13 002A3        BEQL    34$                              ;
                    0C   BE      0C    A8  B1 002A5        CMPW    12(DN), @P                       ; 1505
                                       06  15 002AA        BLEQ    33$                              ;
                                       68  B5 002AC        TSTW    (DN)                             ; 1508
                                       16  12 002AE        BNEQ    37$                              ;
                                       76  11 002B0        BRB     41$                              ; 1509
                                       06  12 002B2 33$:   BNEQ    36$                              ; 1513
                    14   AE            01  D0 002B4 34$:   MOVL    #1, STATUS                       ; 1519
                                       6E  11 002B8 35$:   BRB     41$                              ; 1520
                    0C   AE            08  C0 002BA 36$:   ADDL2   #8, P                            ; 1523
                                 04    AE  D6 002BE        INCL    COUNT                            ; 1524
                                 1C    BE  B6 002C1        INCW    @28(SP)                          ; 1525
```

```
                                  8A    11 002C4          BRB     30$                             ; 1484
                          0C      AE    D5 002C6 37$:     TSTL    P                               ; 1532
                                  0E    12 002C9          BNEQ    38$
                   50     05      A7    9A 002CB          MOVZBL  5(ENTRY), R0                    ; 1533
                   50     07    A047    9E 002CF          MOVAB   7(R0)[ENTRY], R0
 0C     AE         50             01    CB 002D4          BICL3   #1, R0, P
              08   AE             57    D0 002D9 38$:     MOVL    ENTRY, PREV_ENTRY               ; 1534
                                  57    DD 002DD          PUSHL   ENTRY                           ; 1535
           0000V   CF             01    FB 002DF          CALLS   #1, NEXT_REC
                   57             50    D0 002E4          MOVL    R0, ENTRY
        50         57     0C      AE    C3 002E7          SUBL3   P, ENTRY, R0                    ; 1536
                   50             08    C6 002EC          DIVL2   #8, R0
              04   AE             50    C0 002EF          ADDL2   R0, COUNT
                          0C      AE    D4 002F3          CLRL    P                               ; 1537
                                FE5B    31 002F6          BRW     15$                             ; 1372
                          18      BE    95 002F9 39$:     TSTB    @24(SP)                         ; 1547
                                  16    13 002FC          BEQL    40$
                          20      BE    DD 002FE          PUSHL   @32(SP)                         ; 1549
        50    1C   AE             01    C1 00301          ADDL3   #1, 28(SP), R0
                                  50    DD 00306          PUSHL   R0
                   7E     20      BE    9A 00308          MOVZBL  @32(SP), -(SP)
                          1C      AE    DD 0030C          PUSHL   BLOCK
           0000V   CF             04    FB 0030F          CALLS   #4, UPDATE_INDX
                          10      AE    D6 00314 40$:     INCL    BLOCK                           ; 1551
                                  57    D4 00317          CLRL    ENTRY                           ; 1552
                          0C      AE    D4 00319          CLRL    P                               ; 1553
                          04      AE    D4 0031C          CLRL    COUNT                           ; 1554
                   3F     1C      AC    D1 0031F          CMPL    REC_COUNT, #63                  ; 1555
                                  03    1F 00323          BLSSU   41$
                                FDFD    31 00325          BRW     13$
                   03     14      AE    E9 00328 41$:     BLBC    STATUS, 43$                     ; 1567
                                0086    31 0032C 42$:     BRW     47$
                   FA             6E    E9 0032F 43$:     BLBC    MATCHED, 42$                    ; 1568
                          0C      AE    D5 00332          TSTL    P                               ; 1569
                                  7E    12 00335          BNEQ    47$
                          08      AE    D5 00337          TSTL    PREV_ENTRY                      ; 1572
                                  06    13 0033A          BEQL    44$
                   57     08      AE    D0 0033C          MOVL    PREV_ENTRY, ENTRY               ; 1575
                                  54    11 00340          BRB     46$                             ; 1572
                          04      AE    D4 00342 44$:     CLRL    COUNT                           ; 1579
                          10      AE    D7 00345          DECL    BLOCK                           ; 1580
                                  02    DD 00348          PUSHL   #2                              ; 1581
                                  01    DD 0034A          PUSHL   #1
                   50     28      BE    D0 0034C          MOVL    @40(SP), R0
                   51     18      AE    D0 00350          MOVL    BLOCK, R1
                          30    B041    9F 00354          PUSHAB  @48(R0)[R1]
           0000G   CF             03    FB 00358          CALLS   #3, READ_BLOCK
              0C   AE             50    D0 0035D          MOVL    R0, P
              04   A9     0C      AE    D0 00361          MOVL    P, 4(R9)                        ; 1582
                   57     0C      AE    D0 00366 45$:     MOVL    P, ENTRY                        ; 1585
                          0C      AE    DD 0036A          PUSHL   P                               ; 1586
           0000V   CF             01    FB 0036D          CALLS   #1, NEXT_REC
              0C   AE             50    D0 00372          MOVL    R0, P
        50    0C   AE             57    C3 00376          SUBL3   ENTRY, P, R0                    ; 1587
                   51     05      A7    9A 0037B          MOVZBL  5(ENTRY), R1
                                  51    D6 0037F          INCL    R1
                   51             01    8A 00381          BICB2   #1, R1
```

```
                 50           51  C2 00384          SUBL2   R1, R0                          ;
                 50           08  C6 00387          DIVL2   #8, R0                          ;
            04   AE           50  C0 0038A          ADDL2   R0, COUNT                       ;
          FFFF   8F      0C   BE  B1 0038E          CMPW    @P, #65535                      ; 1589
                              D0  12 00394          BNEQ    45$                             ;
                 50           67  3C 00396 46$:     MOVZWL  (ENTRY), R0                     ; 1591
            0C   AE      02 A047  9E 00399          MOVAB   2(R0)[ENTRY], P                 ;
          8000   8F      0C   A8  B1 0039F          CMPW    12(DN), #-32768                 ; 1592
                              0E  12 003A5          BNEQ    47$                             ;
            0C   AE           08  C2 003A7          SUBL2   #8, P                           ; 1595
                         1C   BE  B7 003AB          DECW    @28(SP)                         ; 1596
                         04   AE  D7 003AE          DECL    COUNT                           ; 1597
            14   AE           01  D0 003B1          MOVL    #1, STATUS                      ; 1598
     69     10   AE           01  C1 003B5 47$:     ADDL3   #1, BLOCK, (R9)                 ; 1605
          00D8   CA      04   AE  D0 003BA          MOVL    COUNT, 216(BASE)                ; 1606
            08   A9           57  D0 003C0          MOVL    ENTRY, 8(R9)                    ; 1607
            0C   A9      0C   AE  D0 003C4          MOVL    P, 12(R9)                       ; 1608
            14   A9      08   AE  D0 003C9          MOVL    PREV_ENTRY, 20(R9)              ; 1609
                 50      14   AE  D0 003CE          MOVL    STATUS, R0                      ; 1611
                                  04 003D2          RET                                     ; 1613

; Routine Size:  979 bytes,    Routine Base:  $CODE$ + 0000
```

```
;   626          1614  1 GLOBAL ROUTINE NEXT_REC (ENTRY) : L_NORM =
;   627          1615  1
;   628          1616  1 !++
;   629          1617  1 !
;   630          1618  1 ! FUNCTIONAL DESCRIPTION:
;   631          1619  1 !
;   632          1620  1 !       This routine locates the next directory record and checks it for
;   633          1621  1 !       consistency.
;   634          1622  1 !
;   635          1623  1 !
;   636          1624  1 ! CALLING SEQUENCE:
;   637          1625  1 !       NEXT_REC (ARG1)
;   638          1626  1 !
;   639          1627  1 ! INPUT PARAMETERS:
;   640          1628  1 !       ARG1: address of present record
;   641          1629  1 !
;   642          1630  1 ! IMPLICIT INPUTS:
;   643          1631  1 !       NONE
;   644          1632  1 !
;   645          1633  1 ! OUTPUT PARAMETERS:
;   646          1634  1 !       NONE
;   647          1635  1 !
;   648          1636  1 ! IMPLICIT OUTPUTS:
;   649          1637  1 !       NONE
;   650          1638  1 !
;   651          1639  1 ! ROUTINE VALUE:
;   652          1640  1 !       address of next directory record
;   653          1641  1 !
;   654          1642  1 ! SIDE EFFECTS:
;   655          1643  1 !       NONE
;   656          1644  1 !
;   657          1645  1 !--
;   658          1646  1
;   659          1647  2 BEGIN
;   660          1648  2
;   661          1649  2 MAP
;   662          1650  2         ENTRY           : REF BBLOCK;   ! current directory record
;   663          1651  2
;   664          1652  2 LOCAL
;   665          1653  2         NEXT_ENTRY      : REF BBLOCK;   ! new directory record
;   666          1654  2
;   667          1655  2 BIND_COMMON;
;   668          1656  2
;   669          1657  2 DIR_CONTEXT_DEF;
;   670          1658  2
;   671          1659  2 ! Find the next record by adding in the record size of the current entry.
;   672          1660  2 ! Make sure the record is valid.
;   673          1661  2 !
;   674          1662  2
;   675          1663  2 IF .ENTRY[DIR$W_SIZE] LSSU DIR$C_LENGTH
;   676          1664  2 THEN ERR_EXIT (SS$_BADIRECTORY);
;   677          1665  2 NEXT_ENTRY = .ENTRY + .ENTRY[DIR$W_SIZE] + 2;
;   678          1666  3 IF .NEXT_ENTRY GEQA (.ENTRY + 512 AND NOT 511)
;   679          1667  2 OR .NEXT_ENTRY<0,1>
;   680          1668  2 THEN ERR_EXIT (SS$_BADIRECTORY);
;   681          1669  2
;   682          1670  2 RETURN .NEXT_ENTRY
```

```
;  683          1671  2
;  684          1672  1 END;                                   ! end of routine NEXT_REC



                                     0000 00000          .ENTRY  NEXT_REC, Save nothing                       ; 1614
                      50      00DC   CA   9E 00002          MOVAB   220(BASE), R0                                ; 1653
                      06        04   BC   B1 00007          CMPW    @ENTRY, #6                                   ; 1663
                                     21   1F 0000B          BLSSU   1$                                           ;
                      51        04   BC   3C 0000D          MOVZWL  @ENTRY, R1                                   ; 1665
                      51        04   AC   C0 00011          ADDL2   ENTRY, R1                                    ;
                      51             02   C0 00015          ADDL2   #2, NEXT_ENTRY                               ;
          50      04  AC  00000200   8F   C1 00018          ADDL3   #512, ENTRY, R0                              ; 1666
                      50      01FF   8F   AA 00021          BICW2   #511, R0                                     ;
                      50             51   D1 00026          CMPL    NEXT_ENTRY, R0                               ;
                                     03   1E 00029          BGEQU   1$                                           ;
                      05             51   E9 0002B          BLBC    NEXT_ENTRY, 2$                               ; 1667
                              0828   8F   BF 0002E 1$:      CHMU    #2088                                        ; 1668
                                          04 00032          RET                                                  ;
                      50             51   D0 00033 2$:      MOVL    NEXT_ENTRY, R0                               ; 1670
                                          04 00036          RET                                                  ; 1672

; Routine Size:  55 bytes,    Routine Base:  $CODE$ + 03D3
```

```
;  686          1673  1 GLOBAL ROUTINE UPDATE_INDX (BLOCK, STR_SIZE, STR_ADDR, DIRFCB) : NOVALUE =
;  687          1674  1
;  688          1675  1 !++
;  689          1676  1 !
;  690          1677  1 ! FUNCTIONAL DESCRIPTION:
;  691          1678  1 !
;  692          1679  1 !       This routine updates the indicated cell in the directory file index.
;  693          1680  1 !
;  694          1681  1 !
;  695          1682  1 ! CALLING SEQUENCE:
;  696          1683  1 !       see above
;  697          1684  1 !
;  698          1685  1 ! INPUT PARAMETERS:
;  699          1686  1 !       BLOCK    - zero-based block within directory file
;  700          1687  1 !       STR_SIZE - size of string
;  701          1688  1 !       STR_ADDR - address of string
;  702          1689  1 !       DIRFCB   - address of directory FCB
;  703          1690  1 !
;  704          1691  1 ! OUTPUT PARAMETERS:
;  705          1692  1 !       NONE
;  706          1693  1 !
;  707          1694  1 ! IMPLICIT OUTPUTS:
;  708          1695  1 !       NONE
;  709          1696  1 !
;  710          1697  1 ! SIDE EFFECTS:
;  711          1698  1 !       directory index updated
;  712          1699  1 !
;  713          1700  1 !--
;  714          1701  1
;  715          1702  2 BEGIN
;  716          1703  2
;  717          1704  2 MAP
;  718          1705  2     DIRFCB      : REF BBLOCK;
;  719          1706  2
;  720          1707  2 LOCAL
;  721          1708  2     BLKSPERCELL,
;  722          1709  2     CELL_ADDR,
;  723          1710  2     CELLSIZE,
;  724          1711  2     CELLINDX    : WORD,
;  725          1712  2     DIRINDX     : REF BBLOCK FIELD (DIRC);
;  726          1713  2
;  727          1714  2 IF (DIRINDX = .DIRFCB [FCB$L_DIRINDX]) EQL 0
;  728          1715  2 THEN
;  729          1716  2     RETURN;
;  730          1717  2
;  731          1718  2 IF .DIRINDX [DIRC$W_INUSE] EQL 0
;  732          1719  2 THEN
;  733          1720  3     BEGIN
;  734          1721  3     LOCAL
;  735          1722  3         MAXCELLS;
;  736          1723  3
;  737          1724  3     DIRINDX [DIRC$W_CELLSIZE] = 15;
;  738          1725  3     MAXCELLS = (512 - (DIRINDX [DIRC$T_FIRSTCELL] - DIRINDX [DIRC$W_INUSE]))/15;
;  739          1726  3     BLKSPERCELL = (.DIRFCB [FCB$L_EFBLK]/.MAXCELLS) + 1;
;  740          1727  3     DIRINDX [DIRC$W_BLKSPERCELL] = .BLKSPERCELL;
;  741          1728  3     DIRINDX [DIRC$W_TOTALCELLS] = .DIRFCB [FCB$L_EFBLK]/.BLKSPERCELL;
;  742          1729  3     END
```

```
2 ELSE
2     BLKSPERCELL = .DIRINDX [DIRC$W_BLKSPERCELL];
2
2 IF (.BLOCK + 1) MOD .BLKSPERCELL NEQ 0
2 THEN
2     RETURN;
2
2 CELLINDX = .BLOCK/.BLKSPERCELL;
2
2 IF .CELLINDX GTRU .DIRINDX [DIRC$W_INUSE]
2 THEN
2     RETURN;
2
2 IF .CELLINDX EQL .DIRINDX [DIRC$W_INUSE]
2 THEN
2     DIRINDX [DIRC$W_INUSE] = .DIRINDX [DIRC$W_INUSE] + 1;
2
2 IF .CELLINDX GEQU .DIRINDX [DIRC$W_TOTALCELLS]
2 THEN
2     BUG_CHECK (XQPERR, 'exceeded total number of directory index cells');
2
2 CELLSIZE = .DIRINDX [DIRC$W_CELLSIZE];
2
2 CELL_ADDR = DIRINDX [DIRC$T_FIRSTCELL] + (.CELLINDX)*.(CELLSIZE);
2
2 CH$COPY (.STR_SIZE, .STR_ADDR, 0, .CELLSIZE, .CELL_ADDR);
2
1 END;                                   ! end of routine UPDATE_INDX


                                                  .EXTRN  BUG$_XQPERR

                             003C 00000           .ENTRY  UPDATE_INDX, Save R2,R3,R4,R5        ; 1673
           50         10  AC   D0 00002           MOVL    DIRFCB, R0                           ; 1714
           50       00B0  C0   D0 00006           MOVL    176(R0), DIRINDX                     ;
                          7B   13 0000B           BEQL    5$                                   ;
                          60   B5 0000D           TSTW    (DIRINDX)                            ; 1718
                          2F   12 0000F           BNEQ    1$                                   ;
             04  A0       0F   B0 00011           MOVW    #15, 4(DIRINDX)                      ; 1724
      51         50       50   C3 00015           SUBL3   DIRINDX, DIRINDX, R1                 ; 1725
                 51  01F4 C1   9E 00019           MOVAB   500(R1), R1                          ;
      52         51       0F   C7 0001E           DIVL3   #15, R1, MAXCELLS                    ;
                 51   10  AC   D0 00022           MOVL    DIRFCB, R1                           ; 1726
      51     3C  A1       52   C7 00026           DIVL3   MAXCELLS, 60(R1), R1                 ;
                          51   D6 0002B           INCL    BLKSPERCELL                          ;
             06  A0       51   B0 0002D           MOVW    BLKSPERCELL, 6(DIRINDX)              ; 1727
                 52   10  AC   D0 00031           MOVL    DIRFCB, R2                           ; 1728
      53     3C  A2       51   C7 00035           DIVL3   BLKSPERCELL, 60(R2), R3              ;
             02  A0       53   B0 0003A           MOVW    R3, 2(DIRINDX)                       ;
                          04   11 0003E           BRB     2$                                   ; 1718
                 51   06  A0   3C 00040 1$:       MOVZWL  6(DIRINDX), BLKSPERCELL              ; 1731
 7E      01  04  AC       01   7A 00044 2$:       EMUL    #1, BLOCK, #1, -(SP)                 ; 1733
 52      52      8E       51   7B 0004A           EDIV    BLKSPERCELL, (SP)+, R2, R2           ;
                          52   D5 0004F           TSTL    R2                                   ;
                          35   12 00051           BNEQ    5$                                   ;
      52     04  AC       51   C7 00053           DIVL3   BLKSPERCELL, BLOCK, R2               ; 1737
```

```
                    53           52  B0 00058        MOVW    R2, CELLINDX                              ;
                    60           53  B1 0005B        CMPW    CELLINDX, (DIRINDX)                       ; 1739
                                 28  1A 0005E        BGTRU   5$                                        ;
                    60           53  B1 00060        CMPW    CELLINDX, (DIRINDX)                       ; 1743
                                 02  12 00063        BNEQ    3$                                        ;
                                 60  B6 00065        INCW    (DIRINDX)                                 ; 1745
               02   A0           53  B1 00067 3$:    CMPW    CELLINDX, 2(DIRINDX)                      ; 1747
                                 04  1F 0006B        BLSSU   4$                                        ;
                               FEFF     0006D        BUGW                                              ; 1749
                               0000*    0006F        .WORD   <BUG$_XQPERR!4>                           ;
                    52      04   A0  3C 00071 4$:    MOVZWL  4(DIRINDX), CELLSIZE                      ; 1751
                    51           53  3C 00075        MOVZWL  CELLINDX, R1                              ; 1753
                    51           52  C4 00078        MULL2   CELLSIZE, R1                              ;
                    50      0C A140  9E 0007B        MOVAB   12(R1)[DIRINDX], CELL_ADDR                ;
     52      00     0C  BC      08   AC  2C 00080        MOVC5   STR_SIZE, @STR_ADDR, #0, CELLSIZE, -      ; 1755
                                 60     00087                (CELL_ADDR)                               ;
                                     04 00088 5$:    RET                                               ; 1757

; Routine Size:  137 bytes,    Routine Base:  $CODE$ + 040A
```

```
;  772     1758  1 GLOBAL ROUTINE NEXT_DIR_REC (OLD_REC, VBN) : L_NORM =
;  773     1759  1
;  774     1760  1 !++
;  775     1761  1 !
;  776     1762  1 ! FUNCTIONAL DESCRIPTION:
;  777     1763  1 !
;  778     1764  1 !       This routine advances to the next directory record if the name
;  779     1765  1 !       matches the one given. Note that the directory context pointers
;  780     1766  1 !       are NOT updated.
;  781     1767  1 !
;  782     1768  1 ! CALLING SEQUENCE:
;  783     1769  1 !       DIR_REC (ARG1, ARG2)
;  784     1770  1 !
;  785     1771  1 ! INPUT PARAMETERS:
;  786     1772  1 !       ARG1: address of current directory record
;  787     1773  1 !       ARG2: address of current VBN
;  788     1774  1 !
;  789     1775  1 ! IMPLICIT INPUTS:
;  790     1776  1 !       DIR_FCB: FCB of directory file
;  791     1777  1 !
;  792     1778  1 ! OUTPUT PARAMETERS:
;  793     1779  1 !       ARG2: new VBN if block read
;  794     1780  1 !
;  795     1781  1 ! IMPLICIT OUTPUTS:
;  796     1782  1 !       NONE
;  797     1783  1 !
;  798     1784  1 ! ROUTINE VALUE:
;  799     1785  1 !       address of next record or 0
;  800     1786  1 !
;  801     1787  1 ! SIDE EFFECTS:
;  802     1788  1 !       directory blocks read
;  803     1789  1 !
;  804     1790  1 !--
;  805     1791  1
;  806     1792  2 BEGIN
;  807     1793  2
;  808     1794  2 MAP
;  809     1795  2         OLD_REC         : REF BBLOCK;   ! old directory record
;  810     1796  2
;  811     1797  2 LOCAL
;  812     1798  2         NAME_BUFFER     : VECTOR [FILENAME_LENGTH+1, BYTE],
;  813     1799  2                                         ! buffer to save file name
;  814     1800  2         NEW_REC         : REF BBLOCK;   ! address of new record
;  815     1801  2
;  816     1802  2 BIND_COMMON;
;  817     1803  2
;  818     1804  2 EXTERNAL ROUTINE
;  819     1805  2         READ_BLOCK      : L_NORM;       ! read a disk block
;  820     1806  2
;  821     1807  2
;  822     1808  2 ! Save away the name string of this record. Then advance to the next
;  823     1809  2 ! record, reading the next block if necessary.
;  824     1810  2 !
;  825     1811  2
;  826     1812  2 CH$MOVE (.OLD_REC[DIRSB_NAMECOUNT]+1, OLD_REC[DIRSB_NAMECOUNT], NAME_BUFFER);
;  827     1813  2 NEW_REC = NEXT_REC (.OLD_REC);
;  828     1814  2 IF .NEW_REC[DIRSW_SIZE] EQL 65535
```

```
2 THEN
3     BEGIN
3     .VBN = ..VBN + 1;
3     IF ..VBN GTRU .DIR_FCB[FCB$L_EFBLK]
3     THEN RETURN 0;
3     NEW_REC = READ_BLOCK (..VBN-1 + .DIR_FCB[FCB$L_STLBN], 1, DIRECTORY_TYPE);
2     END;
2
3 IF .NEW_REC[DIR$W_SIZE] + .NEW_REC + 2 GEQA (.NEW_REC + 512 AND NOT 511)
2 OR .NEW_REC[DIR$B_NAMECOUNT] GTRU FILENAME_LENGTH
2 OR .NEW_REC[DIR$B_NAMECOUNT] GTR .NEW_REC[DIR$W_SIZE] + 2 - DIR$C_LENGTH - DIR$C_VERSION
2 THEN ERR_EXIT (SS$_BADIRECTORY);
2
2 ! Compare the name string with the old one. If it matches, return the
2 ! new entry; else 0.
2 !
2
2 IF CH$NEQ (.NEW_REC[DIR$B_NAMECOUNT]+1, NAME_BUFFER,
2            .NEW_REC[DIR$B_NAMECOUNT]+1, NEW_REC[DIR$B_NAMECOUNT])
2 THEN 0
2 ELSE .NEW_REC
2
1 END;                                        ! End of routine NEXT_DIR_REC
```

```
                                 007C 00000        .ENTRY  NEXT_DIR_REC, Save R2,R3,R4,R5,R6       ; 1758
               5E          AC  AE  9E 00002        MOVAB   -84(SP), SP
               56          04  AC  D0 00006        MOVL    OLD_REC, R6                             ; 1812
               50          05  A6  9A 0000A        MOVZBL  5(R6), R0
                               50  D6 0000E        INCL    R0
   6E      05  A6              50  28 00010        MOVC3   R0, 5(R6), NAME_BUFFER
                               56  DD 00015        PUSHL   R6                                      ; 1813
         FF24  CF              01  FB 00017        CALLS   #1, NEXT_REC
               54              50  D0 0001C        MOVL    R0, NEW_REC
         FFFF  8F              64  B1 0001F        CMPW    (NEW_REC), #65535                       ; 1814
                               24  12 00024        BNEQ    1$
                           08  BC  D6 00026        INCL    @VBN                                    ; 1817
               50        00D0  CA  D0 00029        MOVL    208(BASE), R0                           ; 1818
           3C  A0          08  BC  D1 0002E        CMPL    @VBN, 60(R0)
                               50  1A 00033        BGTRU   4$
                               02  DD 00035        PUSHL   #2                                      ; 1820
                               01  DD 00037        PUSHL   #1
   50      08  BC          30  A0  C1 00039        ADDL3   48(R0), @VBN, R0
                           FF  A0  9F 0003F        PUSHAB  -1(R0)
        0000G  CF              03  FB 00042        CALLS   #3, READ_BLOCK
               54              50  D0 00047        MOVL    R0, NEW_REC
               51              64  3C 0004A 1$:    MOVZWL  (NEW_REC), R1                           ; 1823
               52       02 A441    9E 0004D        MOVAB   2(NEW_REC)[R1], R2
               50        0200  C4  9E 00052        MOVAB   512(R4), R0
               50        01FF  8F  AA 00057        BICW2   #511, R0
               50              52  D1 0005C        CMPL    R2, R0
                               12  1E 0005F        BGEQU   2$
          50  8F           05  A4  91 00061        CMPB    5(NEW_REC), #80                         ; 1824
                               0B  1A 00066        BGTRU   2$
```

```
                                    51          0C  C2 00068         SUBL2   #12, R1                              ; 1825
           51      05   A4          08          00  ED 0006B         CMPZV   #0, #8, 5(NEW_REC), R1               ;
                                                05  15 00071         BLEQ    3$                                   ;
                                          0828  8F  BF 00073 2$:     CHMU    #2088                                ; 1826
                                                04  04 00077         RET                                          ;
                                    50      05  A4  9A 00078 3$:     MOVZBL  5(NEW_REC), R0                       ; 1832
                                                50  D6 0007C         INCL    R0                                   ;
                   05   A4          6E          50  29 0007E         CMPC3   R0, NAME_BUFFER, 5(NEW_REC)          ; 1833
                                                03  13 00083         BEQL    5$                                   ;
                                                50  D4 00085 4$:     CLRL    R0                                   ; 1832
                                                    04 00087         RET                                          ;
                                    50          54  D0 00088 5$:     MOVL    NEW_REC, R0                          ; 1835
                                                    04 0008B         RET                                          ; 1837

; Routine Size:  140 bytes,    Routine Base:  $CODE$ + 0493


;  852          1838  1
;  853          1839  1 END
;  854          1840  0 ELUDOM



;                               PSECT SUMMARY
;
;       Name                    Bytes                          Attributes
;
;  $CODE$                         1311  NOVEC,NOWRT, RD , EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                       Library Statistics
;
;                                             -------- Symbols --------       Pages       Processing
;       File                                  Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1              18619       31         0       1000          00:01.8



;                               COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:DIRSCN/OBJ=OBJ$:DIRSCN MSRC$:DIRSCN/UPDATE=(ENH$:DIRSCN)

; Size:           1311 code + 0 data bytes
; Run Time:          00:57.6
; Elapsed Time:      01:51.2
; Lines/CPU Min:      1918
; Lexemes/CPU-Min: 42099
```

```
; Memory Used:  459 pages
; Compilation Complete
```

DEACCS
LIS

DELETE
LIS

DIRSCN
LIS

DISPAT
LIS

CREHDR
LIS

DIRACC
LIS

DELBAD
LIS

CREFCB
LIS

CREWIN
LIS

DISPATCH
LIS

ENTER
LIS

DELFIL
LIS